Unclassified
SECURITY CLASSIFICATION OF THIS PAGE

4

REPORT DOCUMENTATION PAGE

AD-A212 600

1b. RESTRICTIVE MARKINGS

3 DISTRIBUTION / AVAILABILITY OF REPORT
Unlimited

4. PERFORMING ORGANIZATION REPORT NUMBER(S)
TR-89-1037

5. MONITORING ORGANIZATION REPORT NUMBER(S)

6a. NAME OF PERFORMING ORGANIZATION	6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION
Cornell University		Office of Naval Research

6c. ADDRESS (City, State, and ZIP Code)	7b. ADDRESS (City, State, and ZIP Code)
Department of Computer Science Upson Hall, Cornell University Ithaca, NY 14853	800 North Quincy St. Arlington, VA 22217-5000

8a. NAME OF FUNDING / SPONSORING ORGANIZATION	8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
Office of Naval Research		N000014-86-K-0092

8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS			
	PROGRAM ELEMENT NO	PROJECT NO	TASK NO	WORK UNIT ACCESSION NO
800 North Quincy St. Arlington, VA 22217-5000				

11. TITLE (Include Security Classification)
Simpler Proofs for Concurrent Reading and Writing

12. PERSONAL AUTHOR(S)
Fred B. Schneider

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Interim	FROM _____ TO _____	September 13, 1989	5

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	concurrent programming, atomicity, program verification

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Simplified proofs are given for Lamport's protocols to coordinate concurrent reading and writing.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION	
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS		
22a NAME OF RESPONSIBLE INDIVIDUAL Fred B. Schneider	22b. TELEPHONE (Include Area Code) (607) 255-9221	22c. OFFICE SYMBOL

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

Simpler Proofs for Concurrent Reading and Writing*

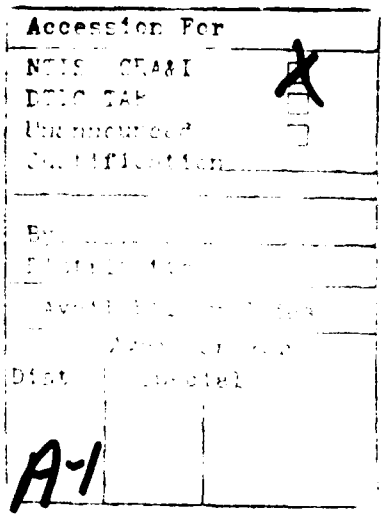
Fred B. Schneider

Department of Computer Science
Cornell University
Ithaca, New York 14853

September 13, 1989

ABSTRACT

Simplified proofs are given for Lamport's protocols to coordinate concurrent reading and writing.



*This material is based on work supported in part by the Office of Naval Research under contract N00014-86-K-0092, the National Science Foundation under Grant No. CCR-8701103, and Digital Equipment Corporation. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author and do not reflect the views of these agencies.

1. Introduction

In most computing systems, hardware ensures that read and write operations to some basic unit of memory can be considered mutually exclusive. As a result, a read that overlaps with a write is serialized and will appear either to precede that write or to follow it. Operations that make multiple accesses to memory are not serialized by the hardware. Therefore, the programmer must ensure that when such operations overlap, they produce meaningful results.

In this paper, we give simplified proofs for some protocols proposed by Lamport [1] to coordinate read and write operations that involve multiple accesses to memory. The two key theorems in [1] are long and intricate. Here, we prove each in only a few lines. Our facility with proofs and the use of formalism in problem solving has improved significantly in a little over 15 years.[1] This is due, in part, to the influence of Edsger Dijkstra.

2. Words from Digits

Consider a computing system in which the basic unit of memory is a *digit*, and a digit can contain one of $S \geq 2$ distinct values. Any finite set of values can be encoded using a fixed set of such digits. We call such a set of digits a *word*. To read the value stored by a word, read operations are performed on some subset of its digits; to write a value, write operations are performed. Observe that overlapping read and write operations to a word will not be serialized by the hardware. Therefore, without additional constraints on execution, it is possible for a read that overlaps a write to obtain a meaningless value. For example, suppose digits can encode integers from 0 through 9, and a word w constructed from three digits initially encodes the value 099. A read that is concurrent with a write of value 100 might obtain any of the following results: 099, 090, 009, 000, 199, 190, 109, 100.

By constraining the order in which digits are read and the order in which digits are written, we can ensure that a read overlapping one or more writes does obtain a meaningful value. Desired are constraints that are both easily implemented and non-intrusive. Execution of neither read nor write operations should be delayed; nor should the constraints require elaborate synchronization primitives.

In the protocols that follow, we consider a word w that is implemented by $n+1$ digits w_0, w_1, ..., w_n. Think of w_0 as the least-significant (right-most) digit and w_n as the most-significant (left-most) digit of a base S number being stored by w. For a digit w_i, define w_i^p to be the value written to w_i by write operation number p.[2] Also define $\mu_i(t)$ to be the number of writes that have been made to digit w_i as of time t. Note that for all i and t, $\mu_i(t) \leq \mu_i(t+1)$.

---

[1] [1] was first submitted for publication in September 1974.

[2] It will be convenient to assume that a write operation to a word writes a value to every digit. The new value can, of course, be the same as the old.

3. Reading to the Left, Writing to the Right

We first show that if the digits of w are read from right to left (i.e. $w_0, w_1, ..., w_n$) but written from left to right (i.e. $w_n, ..., w_1, w_0$) then only certain mixtures of values from overlapping writes are possible. Notice that implementing this constraint on the way digits are accessed delays neither a writer nor a reader.

Lemma 1: If digits of w are written from left to right, then reading the digits from right to left obtains a value $V=w_n^{r_n} \ldots w_1^{r_1} w_0^{r_0}$ such that $r_0 \le r_1 \le \cdots \le r_n$.

Proof: Define t_i such that $r_i=\mu_i(t_i)$. Since digits are read from right to left, $t_0 \le t_1 \le \cdots \le t_n$. For any i, $0 \le i < n$:

$$
\begin{array}{ll}
 & r_i \\
= & \quad \text{«Assumption that } r_i=\mu_i(t_i)\text{»} \\
 & \mu_i(t_i) \\
\le & \quad \text{«Digits are written from left to right»} \\
 & \mu_{i+1}(t_i) \\
\le & \quad \text{«} t_i < t_{i+1} \text{»} \\
 & \mu_{i+1}(t_{i+1}) \\
= & \quad \text{«Assumption that } r_i=\mu_i(t_i)\text{»} \\
 & r_{i+1}
\end{array}
$$

□

Using this result, it is possible to bound the value obtained by a read that overlaps writes to w, provided that the values written to w are non-decreasing. Assume values stored in a word are ordered in the usual lexicographic manner.

Lemma 2: If for all $i \ge 0$, $w_n^i \ldots w_1^i w_0^i \le w_n^{i+1} \ldots w_1^{i+1} w_0^{i+1}$ and $r_0 \le r_1 \le \cdots \le r_n \le r_{n+1}$, then $w_n^{r_n} \ldots w_1^{r_1} w_0^{r_0} \le w_n^{r_{n+1}} \ldots w_1^{r_{n+1}} w_0^{r_{n+1}}$.

Proof: By induction on the number of digits that implement w.

Base Case: $n=0$. By the hypothesis that $r_0 \le r_1 \le \cdots \le r_n \le r_{n+1}$, we conclude $r_0 \le r_{n+1}$. Thus, by the hypothesis that $w_n^i \ldots w_1^i w_0^i \le w_n^{i+1} \ldots w_1^{i+1} w_0^{i+1}$, we have $w_0^{r_0} \le w_0^{r_{n+1}}$.

Induction Step: $n>0$.

$$
\begin{array}{ll}
 & w_n^{r_n} w_{n-1}^{r_{n-1}} \ldots w_1^{r_1} w_0^{r_0} \\
\le & \quad \text{«By induction hypothesis that } w_{n-1}^{r_{n-1}} \ldots w_1^{r_1} w_0^{r_0} \le w_{n-1}^{r_n} \ldots w_1^{r_n} w_0^{r_n} \text{ and lexicographic ordering»} \\
 & w_n^{r_n} w_{n-1}^{r_n} \ldots w_1^{r_n} w_0^{r_n} \\
\le & \quad \text{«By hypothesis that } r_n \le r_{n+1}\text{, and } w_n^i \ldots w_1^i w_0^i \le w_n^{i+1} \ldots w_1^{i+1} w_0^{i+1} \text{»} \\
 & w_n^{r_{n+1}} w_{n-1}^{r_{n+1}} \ldots w_1^{r_{n+1}} w_0^{r_{n+1}}
\end{array}
$$

□

Combining Lemmas 1 and 2 we conclude:

Read-Left, Write-Right: If (i) the sequence of values written to w is non-decreasing, (ii) digits are written from left to right, and (iii) digits are read from right to left, then the value obtained by any read will be no larger than the largest value written by an overlapping write.

There are two interesting things to note about this protocol. First, exclusive access to digits is the only synchronization required. Second, read operations and write operations do not delay each other.

4. Reading to the Right, Writing to the Left

By reversing the order in which digits are read and written, we obtain another protocol for concurrent reading and writing.

Lemma 3: If digits of w are written from right to left, then reading the digits from left to right obtains a value $V=w_n^{r_n} \ldots w_1^{r_1} w_0^{r_0}$ such that $r_n \le \cdots \le r_1 \le r_0$.

Proof: Define t_i such that $r_i=\mu_i(t_i)$. Since digits are read from left to right, $t_n \le \cdots \le t_1 \le t_0$. For any i, $0<i\le n$:

$$
\begin{array}{ll}
 & r_i \\
= & \quad \text{«Assumption that } r_i=\mu_i(t_i)\text{»} \\
 & \mu_i(t_i) \\
\le & \quad \text{«Digits are written from right to left»} \\
 & \mu_{i-1}(t_i) \\
\le & \quad \text{«}t_i<t_{i-1}\text{»} \\
 & \mu_{i-1}(t_{i-1}) \\
= & \quad \text{«Assumption that } r_i=\mu_i(t_i)\text{»} \\
 & r_{i-1}
\end{array}
$$

□

As before, we can bound the value obtained by a read that overlaps writes to w, provided that the values written are non-decreasing. Using Read-Left, Write-Right the value obtained was bounded from above by the largest overlapping write. Having switched the order in which digits are read and written, the value obtained is bounded from below by the smallest overlapping write.

Lemma 4: If for all $i\ge 0$, $w_n^i \ldots w_1^i w_0^i \le w_n^{i+1} \ldots w_1^{i+1} w_0^{i+1}$ and $r_{n+1} \le r_n \le \cdots \le r_1 \le r_0$, then $w_n^{r_n} \ldots w_1^{r_1} w_0^{r_0} \ge w_n^{r_{n+1}} \ldots w_1^{r_{n+1}} w_0^{r_{n+1}}$.

Proof: By induction on the number of digits that implement w.

Base Case: $n=0$. By the hypothesis that $r_{n+1} \le r_n \le \cdots \le r_1 \le r_0$, we conclude $r_{n+1} \le r_n$. Thus, by the hypothesis that $w_n^i \ldots w_1^i w_0^i < w_n^{i+1} \ldots w_1^{i+1} w_0^{i+1}$, we have $w_0^{r_n} \ge w_0^{r_{n+1}}$.

Induction Step: $n>0$.

$$
\begin{array}{ll}
 & w_n^{r_n} w_{n-1}^{r_{n-1}} \ldots w_1^{r_1} w_0^{r_0} \\
\ge & \quad \text{«By induction hypothesis that } w_{n-1}^{r_{n-1}} \ldots w_1^{r_1} w_0^{r_0} \ge w_{n-1}^{r_n} \ldots w_1^{r_n} w_0^{r_n} \text{ and} \\
 & \quad \text{lexicographic ordering»}
\end{array}
$$

$$
\begin{array}{ll}
 & w_n^{r_n}\, w_{n-1}^{r_n} \ldots w_1^{r_n}\, w_0^{r_n} \\
\geq & \quad \text{«By hypothesis that } r_{n+1} \leq r_n \text{, and } w_n^i \ldots w_1^i\, w_0^i \leq w_n^{i+1} \ldots w_1^{i+1}\, w_0^{i+1} \text{»} \\
 & w_n^{r_{n+1}}\, w_{n-1}^{r_{n+1}} \ldots w_1^{r_{n+1}}\, w_0^{r_{n+1}}
\end{array}
$$

□

Combining Lemmas 3 and 4, we conclude:

> **Read-Right, Write-Left**: If (i) the sequence of values written to w is non-decreasing, (ii) digits are written from right to left, and (iii) digits are read from left to right, then the value obtained by any read will be no smaller than any value written by an overlapping write.

As before, exclusive access to digits is the only synchronization required, and operations are never delayed.

5. Conclusion

We have reduced a complicated proof for a subtle protocol to 4 simple lemmas, each consisting of 4 or 5 lines. However, the proof of Lemma 1 is disturbingly similar to the proof of Lemma 3, and the proof of Lemma 2 is disturbingly similar to the proof of Lemma 4. Two proofs should suffice. Perhaps in another 15 years we will find them.

Acknowledgment

David Gries read and commented on an earlier version of this paper.

References

[1] Lamport, L. Concurrent reading and writing. *Comm. ACM 20*, 11 (Nov. 1977), 806-811.